

DATA BASE CONSIDERATIONS FOR VLSI

L.W. Leyking
California Institute of Technology
and Burroughs Corporation

ABSTRACT

This paper will discuss the motivations and history of data base design for design automation. The objectives of using a data base for VLSI design are outlined and the aspects of merging the logical and physical data for a VLSI design are proposed. A hierarchical data base structure and model is presented and typical data examples of logical and physical VLSI data are mapped into this structure.

1.0 Introduction

The issue of data base design for VLSI is of paramount importance. The success of how well designers can model, describe and capture the logical and physical data for a complex chip design will, in part, be measured by the cost and speed of producing new designs in the market place. Companies doing VLSI design will place more emphasis on how fast a product can be brought into the market place to produce a competitive advantage. A well structured design data base will help in achieving this goal.

2.0 Data Base Systems and Definitions

Considerable effort was applied to the data base problem in the mid 60's by the Conference on Data Systems Languages (CODASYL). Work began in 1965 to define the problems and a draft of the work was completed in 1969. Some of the early data base systems were plagued with efficiency problems and lack of capability and it wasn't until the early 70's that most computer manufacturers were offering reliable DBMS as part of their system software. Table 1 charts some of the more common DBMS as of function of the time they were introduced. [1]

Most early DBMS were organized around either a sequential or tree (hierarchical) structure but more recently both network and relational data bases have been proposed and implemented. Figure 1 indicates the four basic types of data structures. Sequential structures consist of records linked together by forward and backward pointers to facilitate tracing from one record to

TABLE # 1

DATA BASE MANAGEMENT SYSTEMS

HISTORY


<div>TIME  (YEAR)</div>																
65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	
BEGIN CODASYL WORK				CODASYL DRAFT		DMS-1100		DMS/90 DBMS/10		DMS 170 IDS 11		DBMS-11		RELATIONAL DBMS		
		IMS	TOTAL		SYSTEM 2000			IBMS SIBAS DMSII		SIMDBM						

FIGURE 1
DATA BASE
RELATIONAL

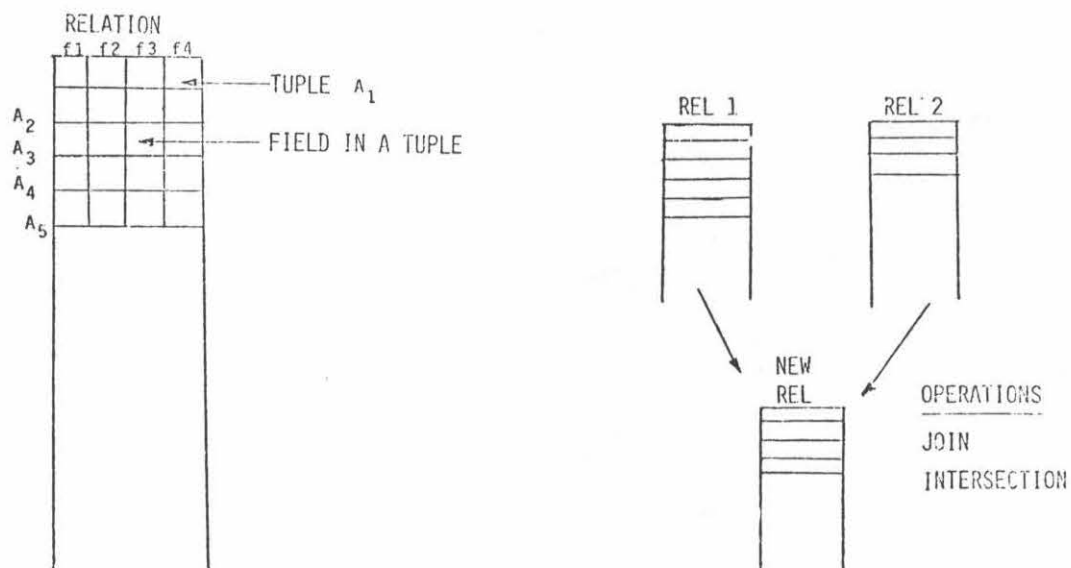
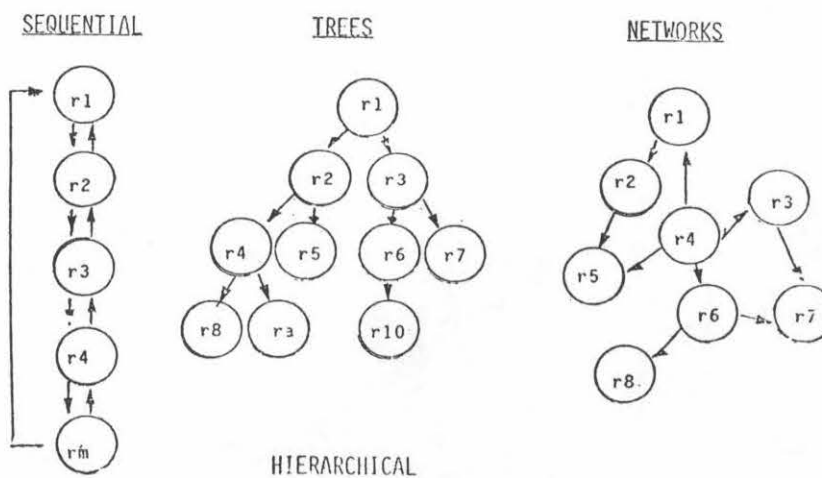


FIGURE 1
DATA BASE
STRUCTURES



another. These records were often sorted on a basic key item which allowed rapid lookup techniques such as binary search. Hashing schemes were often employed to store actual records on magnetic disk in specific pages or segments. Tree structures were used when data organizations dictated a node, branch, and leaf hierarchy with a top down mode of access. Network structures allowed access from any node to any other node in the network and gave the advantages of a tree and sequential structure. Recent data base developments have seen the introduction of relational structures. Individual relations can have basic operations such as join, union, intersection, and projection performed on them to yield new relations. The new relations can be organized to facilitate a particular program's operation. Relational data bases offer flexibility at the cost of efficiency in organizing the data.

Each type of data base structure must be analyzed against the task to be performed. No one structure is best for all problems although in VLSI design automation tasks, the tree and relational structures appear promising.

3.0 Objectives and Motivations of a VLSI Data Base

The need for a VLSI data base is shown by the increased importance VLSI circuit design is taking in the engineering community. A design revolution is underway where more digital computer system functions are being incorporated into VLSI chips, with the end goal of having complete systems on one chip. We are already at this stage with the introduction of microprocessors on a chip and the future indicates even more single chip systems which incorporate more memory and control logic. Today's system chips range in size between the 10K to 30K device (transistor) category with indications of chips approaching 100K devices in a few years. To be able to work with this number of devices, designers will need new data bases to aid in the organization and hierarchical top down structuring of VLSI chip designs.

The following paragraphs outline several major objectives of a VLSI data base, some of which will be essential even in an experimental design system.

1. Hierarchical Structure

To be effective in reducing the amount of design data to a minimum, the design must be hierarchically broken down into smaller and smaller cells which are complete unto themselves. The word complete in this sense means complete in terms of containing all logical and physical data which represent the cell. No additional data should be needed to carry out the VLSI DA design process in terms of simulation, partitioning, interconnection, design rule checking or mask fabrication.

2. Manipulate Large Chips

The data base should be capable of manipulating one or more chips with device numbers approaching 1 million. It is interesting to note how many levels in a hierarchical design are needed to achieve a 1 million device chip. For discussion purposes I will choose a hypothetical design with 5 levels as depicted in Table 2. Level 1 is the highest level cell, the system chip, and level 5 is the lowest level, the primitive cells. Looking at the number of typical subcells/level and the typical number of devices/cell we see that chips between 100K and 2.5M devices can easily be achieved in 5 levels of hierarchy. This number of levels is consistent with the number of levels used in industry to achieve the present day chips. The data base should not be limited in levels, but we do not expect to see 20 levels in the hierarchy, more typically 4-10.

3. Simple to Use

The data base should be simple to use both by the user and the applications programmer. Two sub-objectives could be considered.

A. Using the Data Base Directly

This objective could be desirable as each function program would then not have to build a run time data base by extracting data from the data base but could use the data base directly. Whether this is practical will be determined by the requirements of each function and how it maps to the data base structure. Minimum data redundancy would be achieved using this method.

B. Using the Data Base as a Repository for Data Only

This objective would mean each application program would extract its data from the data base at run time, build its own internal data base for run time efficiency and then deposit any new data back into the data base at run completion. This method would create data redundancy by the application program but would achieve greater run-time efficiency due to the building of special run-time data structures.

4. Non-redundant Data Storage

The data base should store each data item only once in a hierarchical fashion to reduce data storage. Duplication of data will occur in the data base if it is desirable to make explicit data item links between data set records rather than use record pointers. Some data sets will use ordered sets for fast look-up purposes. This technique trades speed of access for data storage but is well justified if rapid data retrieval is necessary.

5. Multiple User Access with Security Protection

The data base should be accessible by many users at one time.

TABLE # 2

SAMPLE CHIP HIERARCHICAL DESIGN LEVELS

<u>LEVEL #</u>	<u>POSSIBLE MODULE DESCRIPTION</u>	<u>RANGE OF SUB MODULES/ LEVEL</u>	<u>DEVICES/ SUBMODULE</u>	<u>TYPICAL TOTAL DEVICES/MODULE</u>	<u>TYPICAL NUMBER SUBMODULES/LEVEL</u>	<u>TYPICAL MODULES</u>
1 Highest	Chip (System)	5 - 50	4K - 100K	100K - 2.5M	25	1
2	Blocks (Processor memory)	5 - 20	400 - 10K	4000 - 100K	10	25
3	Functions (Adders, Counters)	5 - 20	20 - 500	400 - 10K	20	10
4	Cells (4 Bit slice, PLA)	5 - 50	2 - 50	20 - 500	10	20
5 Lowest	Primitive (Group of gates)	-	-	2 - 50	-	10

Note: Each lower level represents the submodules of the level above: i.e. Level 1 (Chips) will have typically 5 - 50 submodule blocks.

Page 4

Most DBMS allow this feature but care must be taken to secure any cell within the data base that is currently being updated by a user. If this is not done, then control over the cells and data items is lost and consistent results cannot be insured.

6. Audit/Recovery

Audit and recovery is a major factor in a production data base to guarantee that data is not lost or destroyed. To implement these features without a software data management system would be very difficult. A reasonable alternative is to copy all files at prescribed intervals for back-up.

7. Update of Data

Each data set record can be assigned a start and stop revision to allow update control. Update control is vital to track a design's history.

4.0 A Logical and Physical Data Model

The key to whether a data base can service a VLSI DA system is how simple, yet general, is its model of the logical and physical system. This section will outline a system model as well as propose a data base structure. Terms related to the data base structure and estimates of the size of the data base records for each data set will also be given.

4.1 System Model

Figure 2 indicates the proposed system model. The basic building block is a cell which is composed of subcells, which themselves can be cells, and primitives which are the bottom or lowest level cells which are not further decomposed.

Each cell is composed of 4 basic parts.

1. Cell Bounding Box (Polygon)
2. Cell primary input and output pins or I/O pins and subcell pins
3. Subcell or primitive references.
4. Net (external and internal) and their associated pins

The primitive has parts 1, 2 and 4 only and does not reference other cells or primitives.

The cell bounding box or polygon is the geometrical enclosure of the physical cell. The cell primary inputs or outputs are the points which join the bounding box to the external nets. The internal nets join the subcell and primitives to other subcells or primitives but do not interface to the external parts of the cell. The subcells are themselves cells which make up the higher level cell. The primitives are the

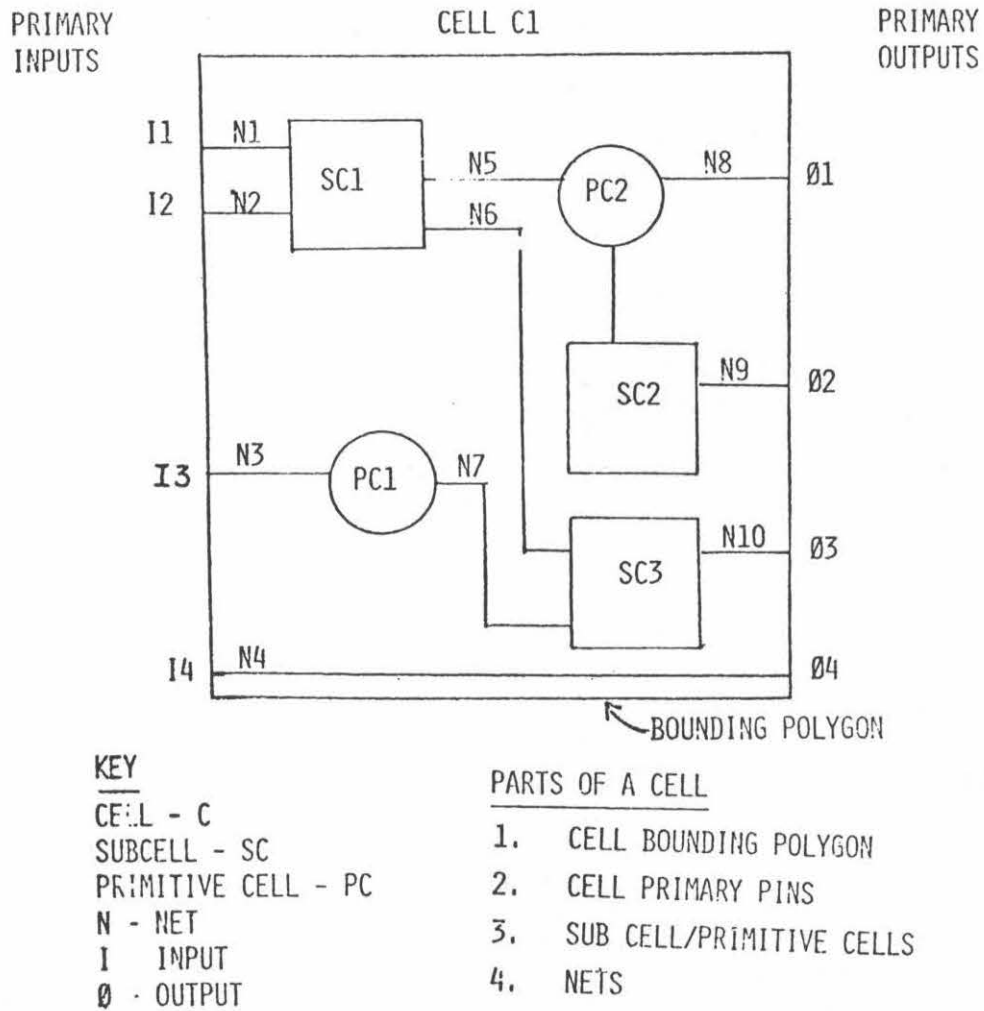
CELL DATA BASE MODEL

Figure 2

lowest level cells which are composed of explicit logic function descriptions and physical geometric descriptions of the primitive. Note that a cell can also be a primitive if it has a logical functional (behavioral) description in terms of its inputs, outputs and states and/or is explicitly defined in terms of its physical geometry.

4.2 Logical and Physical Descriptions

The cell defined in section 4.1 can have both a logical description in terms of its nets (pins and subcells) or its dual model which is subcells (pins and nets) and a physical model which represents the physical geometries of several notations. Each physical description can be thought of as a notational description or representation. Figure 3 indicates 4 possible notations that might be useful to a designer in VLSI work. The logic schematic notation for each of the 4 parts of a cell is shown in terms of its physical descriptions. i.e.: a bounding box, a dot for an I/O pin, a line for a cell net and a geometrical symbol representing the subcells.

Likewise, other notations such as "stick", or electrical have explicit physical descriptions for each piece of a cell. For VLSI work the physical mask notation is most useful as this represents the physical set of rectangles, boxes, and polygons and that make up the geometries of a VLSI mask. Other notations or mixed notations could be capable of being described as long as they contain the 4 basic parts of a cell.

4.3 Our Machine (OM) Hierarchical Description

At this point an example of a hierarchical description of the Caltech, OM data chip is presented. Figure 4 indicates the 4 level hierarchical structure for the "OM" data chip. Shown for each level is the number of cells/level and the estimated number of records/cell. Table 3 indicates an estimate of the total data base records required to store all cells and also indicates the total number of records if all data in all cells was expanded. A 6 to 1 hierarchical data compression ratio results for this "OM" example. Table 4 indicates the estimated number of records for each data set.

5.0 Data Base Structure

Figure 5 outlines a proposed tree data base structure for VLSI DA design. The boxes in the structure represent individual data sets. A data set is a collection of data records in a file usually organized on a set of keys. Index sets, which are collections of records pointing to data set records, are shown as circles and could be applied to each data set for fast look-up purposes. The data base structure is divided into 2 major areas: the logical design and the physical/graphic design. The physical design is complementary to the logical in the sense that each

SAMPLE ENGINEERING NOTATIONS







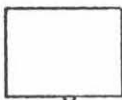

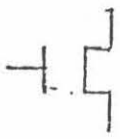
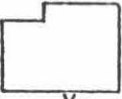

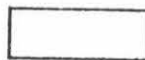
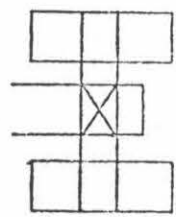
NOTATION TYPE	MODULE BOUNDING BOX	MODULE PINS	MODULE NETS	PRIMITIVES SYMBOLS
LOGIC	Y  X RECT.	LOGIC PINS	 LINES	
STICK	Y  X RECT.	INTERSECTIONS	 LINES	
ELECTRICAL	Y  X RECT.	DOTS	 LINES	
PHYSICAL	Y  X POLYGON	 CONTACTS	 SHEETS	 SET OF SHEETS

Figure 3

FIGURE #4

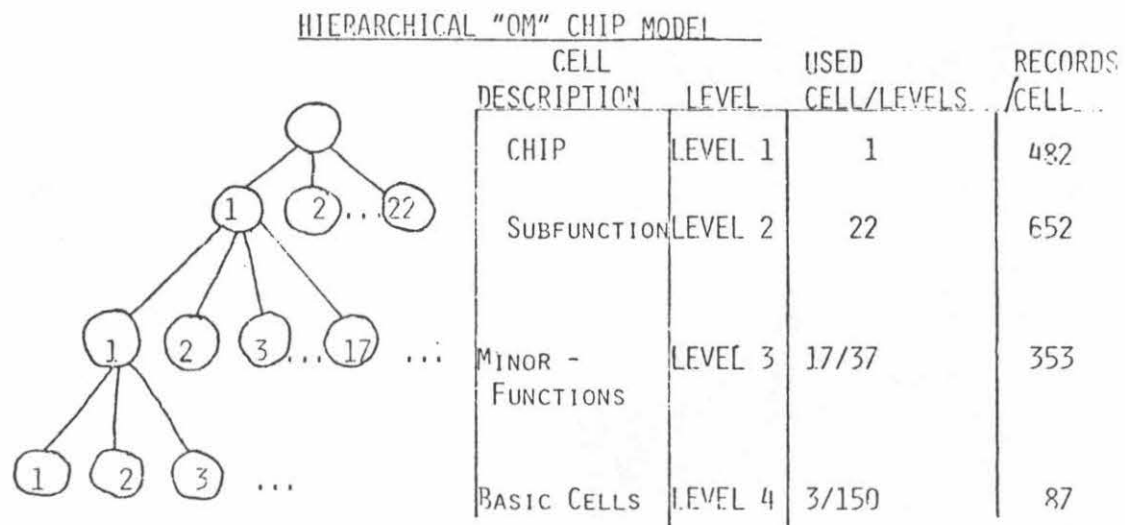


TABLE #3
 "OM" DATA BASE RECORDS

LEVEL	TOTAL EXPLODED RECORDS/LEVEL		TOTAL DATA BASE RECORDS	
1	(1x 482)	482	(1 x 482)	482
2	(22 x 652)	14344	(22 x 652)	14344
3	(22 x 17 x 353)	132022	(37 x 353)	13061
4	(22 x 17 x 3 x 87)	97614	(150 x 87)	13050
	TOTAL	244462	TOTAL	40937

DATA COMPACTION RATIO: 6:1

TABLE #4

ESTIMATED RECORDS / LEVEL FOR "OM" CHIP

NUMBER OF LEVELS FOR CHIP >>	4
DIFFERENT SUBMODULES AT LEVEL 1 >>	22
SUBMODULES PER MODULE AT LEVEL 1 >>	22
NETS PER MODULE AT LEVEL 1 >>	100
PINS PER MODULE AT LEVEL 1 >>	250
GT PER MODULE AT LEVEL 1	100
RULES PER MODULE AT LEVEL 1 >>	10
DIFFERENT SUBMODULES AT LEVEL 2 >>	37
SUBMODULES PER MODULE AT LEVEL 2 >>	17
NETS PER MODULE AT LEVEL 2 >>	150
PINS PER MODULE AT LEVEL 2 >>	375
GT PER MODULE AT LEVEL 2 >>	100
RULES PER MODULE AT LEVEL 2 >>	10
DIFFERENT SUBMODULES AT LEVEL 3 >>	150
SUBMODULES PER MODULE AT LEVEL 3 >>	3
NETS PER MODULE AT LEVEL 3 >>	70
PINS PER MODULE AT LEVEL 3 >>	170
GT PER MODULE AT LEVEL 3 >>	100
RULES PER MODULE AT LEVEL 3 >>	10
DIFFERENT SUBMODULES AT LEVEL 4 >>	0
SUBMODULES PER MODULE AT LEVEL 4 >>	0
NETS PER MODULE AT LEVEL 4 >>	10
PINS PER MODULE AT LEVEL 4 >>	27
GT PER MODULE AT LEVEL 4 >>	40
RULES PER MODULE AT LEVEL 4 >>	10

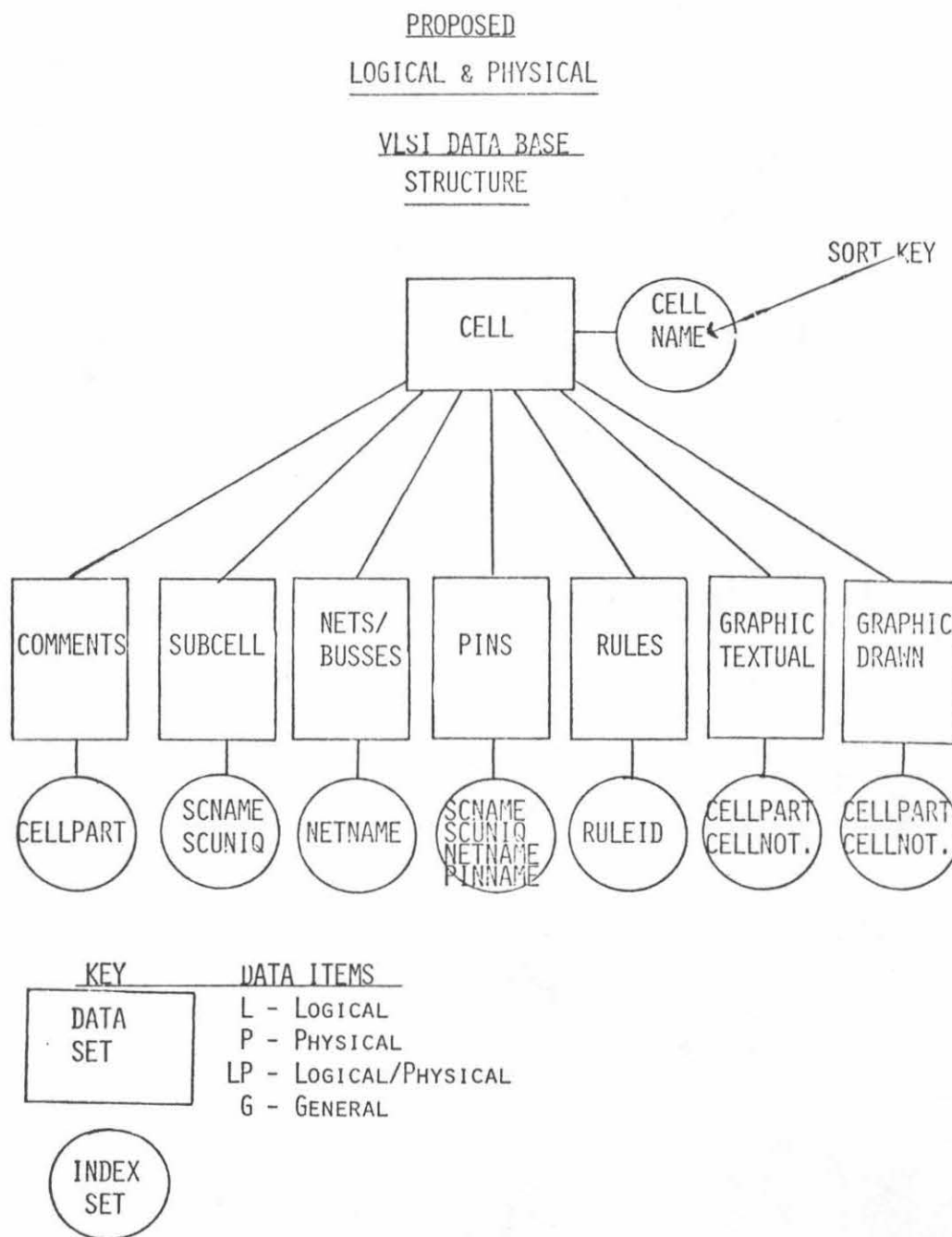


FIGURE 5

logical quantity relative to a cell contains one or more physical descriptions i.e. : nets have physical pad sizes and shapes, etc. Treating each physical quantity in the data base as a part of the notation system, allows multiple physical descriptions for one logical description.

5.1 Cell Data Set

The cell logical data set contains all the global data items, such as number of subcells, in the cell or whether a cell is a primitive. This top level data set is the entry point to each user program in that all subcells which are of interest can be derived in a top down fashion from the highest level cell. Only those levels in the design hierarchy which are of interest need be accessed. For instance, if a simulation of a VLSI chip at its next lowest level is needed, then only 1 level down in the hierarchy need be accessed. Figure 6 indicates a block diagram of the OM data chip. The chip is broken down into 22 subcells. If a logic primitive functional description was written for each of these subcells in terms of its input, output and states, then the chip could be simulated in terms of its 22 pieces. Likewise, if a logic primitive functional description was written for the chip, then this description could be simulated and compared against the simulation of its 22 pieces to verify consistency of the logical design. This technique can be a useful in design verification and the data base can house the logical functional description in the rules data set.

5.2 Subcell Data Set

The subcell logic data set will hold all the references to subcells or primitives within a given cell. Hierarchy is developed by iterating back to the cell level for each subcell to find its associated pieces.

5.3 Comment Text Data Set

This data set is intended to be a user comment text file. Possible uses for this data set could be to store cell specifications at the next higher level, user documentation of the design or design status of the cell.

5.4 Nets/Busses Data Set

The nets/busses logic data set will contain all nets inside a cell. Net types can be external, those touching the cell bounding box or internal, those which do not connect to external I/O pins.

5.5 Pins Data Set

The pins logic data set will house all the pin related data in each cell and the associated explicit pin information for each

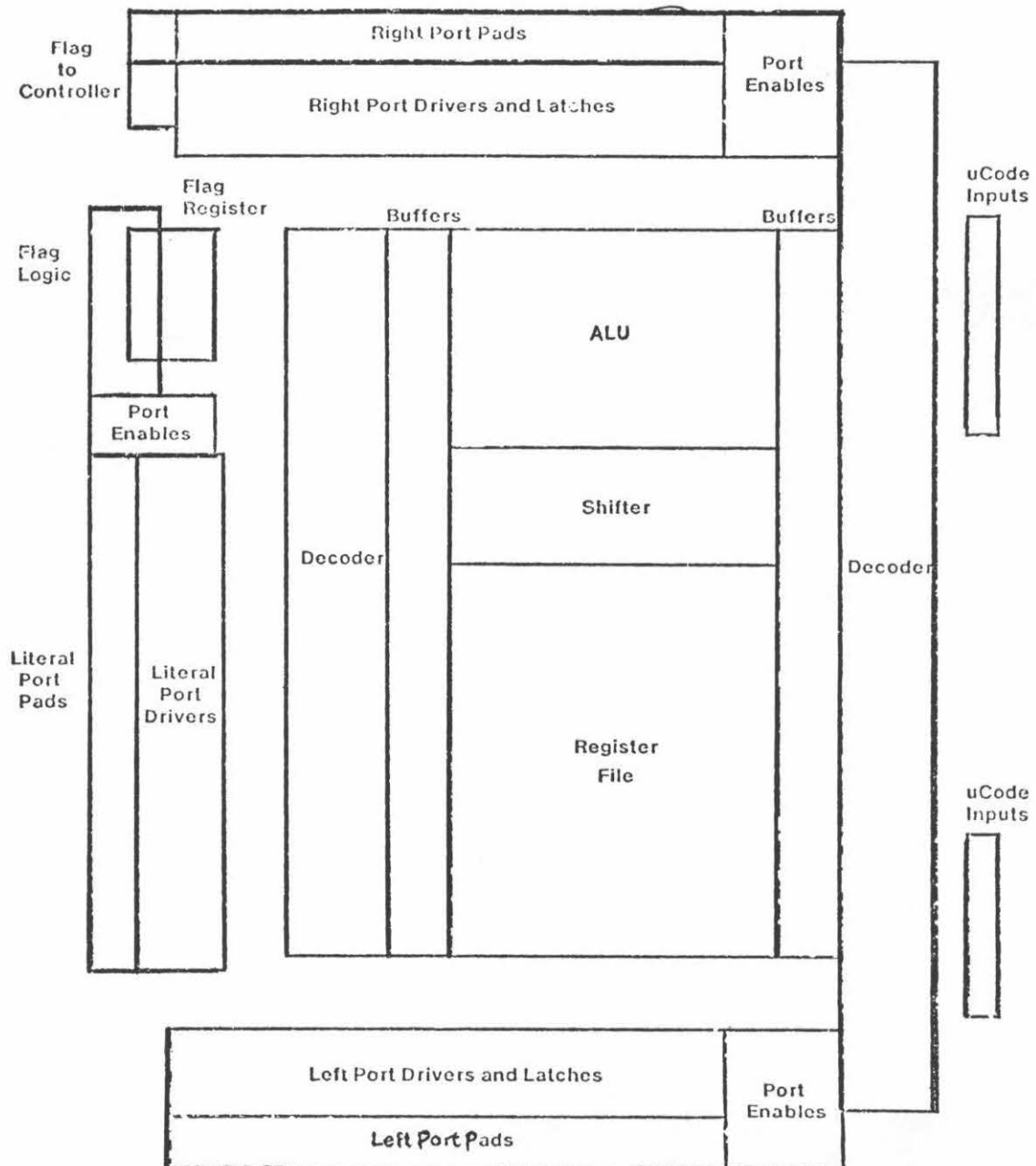


Figure 6 Map Locating the Major Blocks of the Datachip

subcell within a cell.

5.6 Rules Data Set

The rules logical and physical data set would allow a card image description of the rules for each level of notation the user wishes to employ. The language for each of these rules notations can be natural to that notation. For example, the mask physical notation would employ a design rule check language which would outline the design rules for inter and intra layer spacings. The electrical notation would house rules similar to those for circuit simulators such as MSINC or SPICE and could contain exactly those languages. The "stick" notation could contain spacing rules relative to each line in a stick drawing and the logic notation rules could contain the functional description of the primitive in terms of its inputs, outputs and state.

5.7 Graphic Textual Data Set

This data set houses the graphic textual data for one of the 4 physical parts of a cell. A graphic textual language could be standardized across all notation types or could be specific to a notation. Languages similar to CIF [2], and ICLIC [3] could be used to describe the physical geometries of a cell.

5.8 Graphic Drawn Data Set

The graphic drawn data set stores the user drawn data representing one of the 4 physical parts of the cell. A user could sit in front of an interactive storage or refresh tube and physically draw a primitive structure, a net track, a cell bounding box or a cell pin and expect this polygon data to be stored for later use. Figure 7 indicates graphic textual and graphic drawn data. Methods for translating from graphic textual data to graphic drawn data are well known since this is a one-to-many transformation. Translating from graphic drawn to graphic textual is more difficult since this is a many-to-one transformation requiring some form of pattern recognition. Recognition of step and repeat patterns would be very difficult in this reverse translation. The pattern recognition process is needed in VLSI design for determining a transistor or gate from a set of polygons.

5.9 Data Base Size Estimates

One important aspect of the proposed data base is how large will it become for a chip of 100K devices. Tables 5 and 6 relate to the data shown in Table 2 and indicates data set sizes for each of the data set types described. The key concept here is the hierarchical structure of the data base and allowing necessary data to be carried only once in any one cell or primitive.

Assumptions in the calculation of set sizes:

FIGURE 7

GRAPHIC TEXTUAL DESCRIPTION

POLYGON X1 Y1 X2 Y2 X3 Y3 X4 Y4;

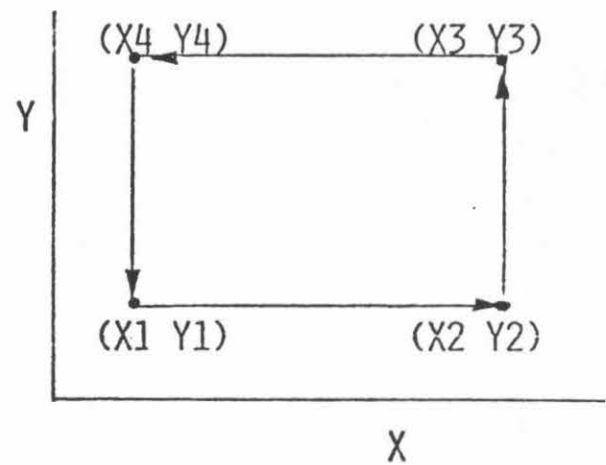
GRAPHIC DRAWN DESCRIPTION

TABLE 5
ESTIMATES OF DATA SET SIZES IN A VLSI DATA BASE*
FOR 100K DEVICE CHIP

DATA SETS	ESTIMATED RECORD SIZE (CHARS.)	ESTIMATED TOTAL RECORDS FOR 100K DEVICE CHIP	TOTAL CHARS.
CELLS & PRIMITIVES	189	66	12,474
COMMENT TEXT	98	1320	129,360
NETS	39	13200	514,800
PINS	64	35640	2,280,960
SUBCELLS	53	1320	69,960
RULES	90	4500	405,000
GRAPHIC TEXTUAL	98	8000	784,000
GRAPHIC DRAWN	20	16000	320,000
TOTALS	661	80046	4,516,554 CHARACTERS

* INITIAL ESTIMATES ONLY - NOT DEEMED ACCURATE

TABLE 6
ASSUMED AVERAGE SIZE OF EACH CELL

DATA BASE ITEM	QUANTITIES
COMMENT TEXT	20 LINES
NETS/BUSSES	200
PINS	$2.5/\text{NETS} + 40/\text{CELL} = 540$
SUBCELLS	20/CELL
RULES	200 LINES/PRIMITIVE, 20 LINES/MODUL
GRAPHIC TEXTUAL	500 LINES/PRIMITIVE, 100 LINES/MODU
GRAPHIC DRAWN	1000 LINES/PRIMITIVE, 200 LINES/MODUL

INVERTER CELL (IC)

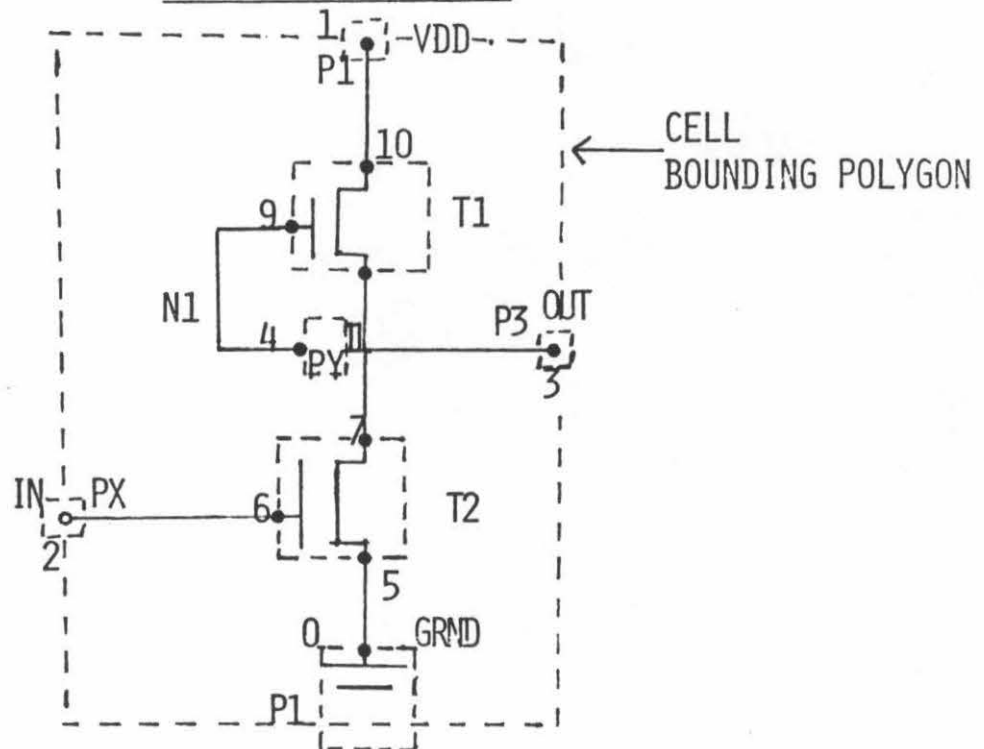


FIGURE 8

INVERTER CELLDATA BASE DATA

DATA SET	DATA			
CELL	CELLNAME (INVERTER CELL - IC) INPUTS (1,2) OUTPUTS (0,3) PRIMITIVE (NO)			
SUBCELLS	SUBCELLNAME	SUBCELL UNIQ	SUBCELL LOG	COMMENT
	P	1	-3, -2	PAD
	P	2	6, 22	PAD
	P	3	6,-10	PAD
	PX	1	-2,-10	PAD
	PY	1	0, 13	PAD-DOUBLE
	T1	1	5.5, 17	DEPL TRAN
	T2	1	1, -3	ENH TRAN
	TABLE 7			

INVERTER CELLDATA BASE DATA

DATA SET		DATA		
	NETNAME	NETCLASS	# PINS	PINS
NETS	VDD	POWER	2	1,10
	OUT	EXTERNAL LOGIC	4	3,7,8,11
	IN	LOGIC	2	2,6
	GRND	POWER	2	0,5
	N1	INTERNAL LOGIC	2	9,4
	CELLNAME	UNIQ	NETNAME	PINID
PINS	IC	1	GRND	0
	IC	1	VDD	1
	IC	1	IN	2
	IC	1	OUT	3
	T1	1	VDD	10
	T1	1	N1	9
	T1	1	OUT	8
	T2	1	OUT	7
	T2	1	IN	6
	T2	1	GRND	5
	P	1	GRND	0
	P	2	VDD	1
	P	3	OUT	3
	PX	1	IN	2
	PY	1	N1	4
	PY	1	OUT	11
TABLE 8				

INVERTER CELLDATA BASE DATA

DATA SET	DATA		
GRAPHIC TEXT	CELL NOTATION	CELL PART	TEXT
	PHYSICAL POLYGONS (MASK)	CELL BOUNDING POLYGON (1)	POLYGON ((-2,-10), (2,-10)...);
		CELL PRIMARY PINS (2)	CALL P1; CALL P2; CALL P3; CALL PX;
		↓	
		SUBCELLS (3)	CALL PY; CALL T1; CALL T2;
		↓	
		NETS (4)	VDD - NONE OUT - POLYGON ((7,-6), (9,-6), (9,17), (8,17), (8,14), (7,14), (7,-6)), IN - POLYGON ((0,-6), (2,-6), (2,-5) (5,-5), (5,-3), (0,-3), (0,-6)); GRND - (NONE) N1 - POLYGON((3,18), (5.5,18), (5.5,21 (3,21), (3,18));
	↓	↓	

TABLE 9

INVERTER CELLDATA BASE DATA

DATA SET	DATA	
RULES	RULE ID	RULES
	1. FUNCTIONAL SIMULATION	OUT = $\overline{\text{IN}}$; DELAY = IONS;
	2. MSINC ELECTRICAL SIMULATION	T1 = BDEP; 1 λ , 1 λ ; T2 = BENH; 4 λ , 1 λ ; TOL = .01 VDD = 5VOLTS; VIN = 5,400N,0, 410N; TIME = 10N, 300N; PLOT = VOUT (0,3), VIN (0,2);
	3. DESIGN RULE CHECK	DIFFUSION = 2 λ ; POLY = 2 λ ; POLY TO DIFFUSION = 1 λ ; METAL = 3 λ METAL TO METAL = 3 λ ; ETC.
	4. DIMENSIONAL ETC.	$\lambda = 2$

TABLE 10

1. Only 5 levels in hierarchy for sample hypothetical chip.
2. Primitives composed of 2 - 50 transistors
3. Number of typical subcells at each level in hierarchy as shown in Table 2.

Table 5 points out that over 4 1/2 million characters of storage would be needed for a chip of 100K devices.

6.0 Sample Data Mapping of an Inverter Cell into Proposed Data Base

Figures 8 and 9 indicate simple inverter cell electrical and mask drawings. Each subcell and node is numbered to allow a net list to be constructed.

Tables 7-10 indicate the outline of the data base data for each selected data set as described earlier.

It would appear that a simple data base model and structure as described, used in a hierarchical fashion, would suffice for a Design Automation System.

References

- [1] Computing Surveys, Volume 8, 1976, ACM
- [2] Introduction to VLSI Systems, C.A. Mead, L.A. Conway
Chapter 4, Section 2, To be published
- [3] A Language Processor and a Sample Language
Thesis - R. Ayres, June 12, 1978, California Institute of Technology